Práctico 4
Introducción a la programación imperativa (continuación)

Algoritmos y Estructuras de Datos I

1. Anotar con predicados los siguientes programas utilizando el transformador de predicados *wp*.

   (a) Var $x, y : Num$
   $\{\quad\}$
   $x := x + y;$
   $\{x = 6 \wedge y = 5\}$

   (b) Var $x : Num$
   $\{\quad\}$
   $x := 8$
   $\{x = 8\}$

   (c) Var $x : Num$
   $\{\quad\}$
   $x := 8$
   $\{x = 7\}$

   (d) Var $x, y : Num$
   $\{\quad\}$
   $x, y := y, x$
   $\{x = B \wedge y = A\}$

   (e) Var $x, y, a : Num$
   $\{\quad\}$
   $a, x := x, y$
   $\{\quad\}$
   $y := a$
   $\{x = B \wedge y = A\}$

   (f) Var $x, y : Num$
   $\{\quad\}$
   if $x \geq y \rightarrow$
   $\{\quad\}$
   $x := 0$
   $\{\quad\}$
   $\square$ $x \leq y \rightarrow$
   $\{\quad\}$
   $x := 2$
   $\{\quad\}$
   fi
   $\{(x = 0 \vee x = 2) \wedge y = 1\}$

2. Demostrar que los siguientes programas anotados (ternas de Hoare) son correctos. En todos los casos las variables x, y son de tipo Int, a, b de tipo $Bool$ y N es una constante de tipo Int.

   (a) $\{True\}$
   if $x \geq 1 \rightarrow x := x + 1$
   $\square$ $x \leq 1 \rightarrow x := x - 1$
   fi
   $\{x \neq 1\}$

   (b) $\{x \neq y\}$
   if $x > y \rightarrow$ **skip**
   $\square$ $x < y \rightarrow x, y := y, x$
   fi
   $\{x > y\}$

   (c) $\{True\}$
   $x, y := y * y, x * x;$
   if $x \geq y \rightarrow x := x - y$
   $\square$ $x \leq y \rightarrow y := y - x$
   fi
   $\{x \geq 0 \wedge y \geq 0\}$

   (d) $\{True\}$
   if $\neg a \vee b \rightarrow a := \neg a$
   $\square$ $a \vee \neg b \rightarrow b := \neg b$
   fi
   $\{a \vee b\}$

   (e) $\{N \geq 0\}$
   $x := 0;$
   do $x \neq N \rightarrow x := x + 1$
   od
   $\{x = N\}$

   (f) $\{True\}$
   $r := N;$
   do $r \neq 0 \rightarrow$
   if $r < 0 \rightarrow r := r + 1$
   $\square$ $r > 0 \rightarrow r := r - 1$
   fi
   od
   $\{r = 0\}$

3. Calcular las expresiones **E** y **F** de modo que los siguientes programas sean correctos:

   (a) Var $x, y : Nat$
   $\{True\}$
   $x, y := x + 1, \mathbf{E}$
   $\{y = x + 1\}$

   (b) Var $a, q, c, w : Num$
   $\{q = a * c \wedge w = c^2\}$
   $a, q := a + c, \mathbf{E}$
   $\{q = a * c\}$

   (c) Const $A, B : Nat$
   Var $q, r : Nat$
   $\{A = q * B + r\}$
   $q := \mathbf{E};\ r := r - B$
   $\{A = q * B + r\}$

   (d) Const $N : Num$
   Var $x, y, p, q : Num$
   $\{x * y + p * q = N\}$
   $x := x - p;$
   $q := \mathbf{F}$
   $\{x * y + p * q = N\}$

4. Especifique los siguientes problemas con ternas de Hoare y luego derive un programa que los verifique.

(a) Calcular el mínimo de dos valores.

(b) Calcular el valor absoluto de un número.

5. Demostrar que si el programa

$$\begin{array}{l}\{P\}\\ \textbf{if } B_0 \to S_0\\ \ \square\ B_1 \to S_1\\ \textbf{fi}\\ \{Q\}\end{array} \quad \text{es correcto, entonces también lo es} \quad \begin{array}{l}\{P\}\\ \textbf{if } B_0 \to S_0\\ \ \square\ \neg B_0 \to S_1\\ \textbf{fi}\\ \{Q\}\end{array}$$

¿Qué utilidad tiene esta propiedad cuando se programa en lenguaje C?

6. Explicar en lenguaje natural que piden las especificaciones de los siguientes programas y analizar qué hace realmente cada uno. En base a estas observaciones determinar si son correctos.

(a)
Const $N : Num$
Var $s, i : Num$
$a : array[0, N)\ of Num$
$\{N \geq 0\}$
$i, s := 0, 0$
do $i \neq N \to$
$s := s + a.i$
od
$\{s = \langle \sum k\ : 0 \leq k < N :\ a.i \rangle\}$

(b)
Const $N : Num$
Var $s, i : Num$
$a : array[0, N)\ of Num$
$\{N \geq 0\}$
$i, s := 0, 0$
do $i \neq N \to$
$i := i + 1$
$s := s + a.i$
od
$\{s = \langle \sum k\ : 0 \leq k < N :\ a.i \rangle\}$

(c)
Const $N : Num$
Var $s, i : Num$
$a : array[0, N)\ of Num$
$\{N \geq 0\}$
$i, s := -1, 0$
do $i \neq N \to$
$i := i + 1$
$s := s + a.i$
od
$\{s = \langle \sum k\ : 0 \leq k < N :\ a.i \rangle\}$

(d)
Const $N : Num$
Var $i : Num\ r : Bool$
$a : array[0, N)\ of Num$
$\{N \geq 0\}$
$i, r := 0, False$
do $i \neq N \wedge \neg r \to$
if $a.i = e \to r := True$
$\square\ a.i \neq e \to$ **skip**
fi
$i := i + 1$
od
$\{\langle \exists k\ : 0 \leq k < N :\ a.k = e \rangle \Rightarrow a.i = e\}$

7. Decidir y justificar si los siguientes predicados son invariantes para el programa 6.(b):

(a) $\{1 \leq i\}$

(b) $\{0 \leq i\}$

(c) $\{0 \leq i \leq N\}$

(d) $\{s = \langle \sum k\ : 0 \leq k < N :\ a.i \rangle\}$

(e) $\{0 \leq s \leq \langle \sum k\ : 0 \leq k < N :\ a.i \rangle\}$

8. Derivar los siguientes programas con bucles utilizando los invariantes dados:

(a) Desarrollar un algoritmo para calcular el máximo común divisor entre dos enteros positivos, especificado como:
Const $X, Y : Int$
Var $x, y : Int$
$\{X > 0 \wedge Y > 0 \wedge x = X \wedge y = Y\}$
S
$\{x = mcd.X.Y\}$
Utilizando como invariante $\{I : x > 0 \wedge y > 0 \wedge mcd.x.y = mcd.X.Y\}$

Para derivar S serán de utilidad las propiedades del mcd:
(1) $mcd.x.x = x$
(2) $mcd.x.y = mcd.y.x$
(3) $x > y\ \Rightarrow\ mcd.x.y = mcd.(x - y).y$
$y > x\ \Rightarrow\ mcd.x.y = mcd.x.(y - x)$

(b) Derivar dos programas que calculen $r = X^Y$ a partir de cada una de las siguientes definiciones funcionales de la función exponencial especificada como $exp.x.y = x^y$:

(i) $$\begin{array}{lll} exp.x.y & = & (\ y = 0 \to 1 \\ & & \Box\ y \neq 0 \to x * exp.x.(y-1) \\ & &) \end{array}$$

(ii) $$\begin{array}{lll} exp.x.y & = & (\ y = 0 \to 1 \\ & & \Box\ y \neq 0 \to (\ y \bmod 2 = 0 \to exp.(x*x).(y \div 2) \\ & & \qquad \Box\ y \bmod 2 = 1 \to x * exp.x.(y-1) \\ & & \qquad) \\ & &) \end{array}$$

Diseñar los dos programas a partir de:

Precondición R: $\{x = X \wedge y = Y \wedge x \geq 0 \wedge y \geq 0\}$

Postcondición Q: $\{r = X^Y\}$

Invariante I: $\{y \geq 0 \wedge r * x^y = X^Y\}$

Para cada programa usar una de las definiciones. Tener en cuenta las mismas a la hora de decidir la manera de achicar la cota.

Ejercicios extra

7. Demostrar que si el programa

$$\begin{array}{l} \{P\} \\ \textbf{if } B \to S \\ \textbf{fi} \\ \{Q\} \end{array} \qquad \text{es correcto, entonces también lo es} \qquad \begin{array}{c} \{P\} \\ \text{S} \\ \{Q\} \end{array}$$

8. *Calcular* una precondición P de modo que sean correctos los siguientes programas anotados. Agregar además las anotaciones intermedias en caso que haya sentencias compuestas con ";". Suponer que las variables x, y, z, q, r son de tipo Int, las variables i, j de tipo Nat y las variables a, b de tipo $Bool$:

(a) $\{P\}\ x := 8\ \{x = 8\}$

(b) $\{P\}\ x := 8\ \{x \neq 8\}$

(c) $\{P\}\ x := 9\ \{x = 7\}$

(d) $\{P\}\ x := x + 1;\ y := y - 2\ \{x + y = 0\}$

(e) $\{P\}\ x := x + 1;\ y := y - 1\ \{x * y = 0\}$

(f) $\{P\}\ x := x + 1;\ y := y - 1\ \{x + y + 10 = 0\}$

(g) $\{P\}\ z := z * y;\ x := x - 1\ \{z * y^x = C\}$

(h) $\{P\}\ x, y, z := 1, d, c\ \{x * x^y = c^d\}$

(i) $\{P\}\ i, j := i + i, j;\ j := j + i\ \{i = j\}$

(j) $\{P\}\ x := (x - y) * (x + y)\ \{x + y^2 = 0\}$

(k) $\{P\}\ q, r := q + 1, r - y\ \{q * y + r = x\}$

(l) $\{P\}\ a := a \equiv b;\ b := a \equiv b;\ a := a \equiv b\ \{(a \equiv B) \wedge (b \equiv A)\}$

9. (a) Usando las propiedades del transformador de predicados *weakest precondition* que se encuentran en los puntos 1 y 9 del "Digesto para la programación imperativa", demostrar las siguientes propiedades:

I. $\{P\}\ S\ \{Q\} \wedge [P_0 \Rightarrow P] \ \Rightarrow\ \{P_0\}\ S\ \{Q\}$

II. $\{P\}\ S\ \{Q\} \wedge [Q \Rightarrow Q_0] \ \Rightarrow\ \{P\}\ S\ \{Q_0\}$

III. $\{P\}\ S\ \{Q\} \wedge \{P\}\ S\ \{R\} \ \equiv\ \{P\}\ S\ \{Q \wedge R\}$

IV. $\{P\}\ S\ \{Q\} \wedge \{R\}\ S\ \{Q\} \ \equiv\ \{P \vee R\}\ S\ \{Q\}$

(b) Desde un punto de vista práctico, ¿qué aportan las propiedades anteriores a la hora de verificar la corrección de un programa?

10. Dado $n > 0$, desarrollar un programa que devuelva en la variable k la mayor potencia de 2 menor o igual que n, utilizando el invariante dado.

Precondición R: $\{n > 0\}$

Postcondición Q: $\{0 < k \leq n \wedge n < 2 * k \wedge \langle\, \exists j\ : 0 \leq j :\ k = 2^j \,\rangle\}$

Invariante I: $\{0 < k \leq n \wedge \langle\, \exists j\ : 0 \leq j :\ k = 2^j \,\rangle\}$

11. Decimos que dos programas S y T son *equivalentes*, denotado por $S \simeq T$, si y solo si $wp.S.Q \equiv wp.T.Q$ para cualquier predicado Q. ¿Cuál es la intuición que justifica esta definición?

12. Sean S, S_0, S_1, T programas cualesquiera, B_0, B_1 guardas cualesquiera, E, F expresiones cualesquiera. En cada caso, ¿son equivalentes los programas *i*, *ii* e *iii*? En caso afirmativo demostrarlo, en caso negativo dar un contraejemplo (instanciando los programas y las guardas).

(a) *i)*
$x := E;$
$y := F;$

ii)
$y := F;$
$x := E;$

(b) *i)*
if $B_0 \to S$
$\square\ B_1 \to S$
fi

ii) S

(c) *i)*
if $B_0 \to S; S_0; T$
$\square\ B_1 \to S; S_1; T$
fi

ii)
if $B_0 \to S; S_0$
$\square\ B_1 \to S; S_1$
fi;
T

iii)
S;
if $B_0 \to S_0; T$
$\square\ B_1 \to S_1; T$
fi

13. Considerar los siguientes programas anotados, donde la variable x es de tipo Int:

i)
$\{P\}$
do $x \neq 0 \to x := x - 1$
od
$\{x = 0\}$

ii)
$\{P\}$
do $x \neq 0 \to x := x - 2$
od
$\{x = 0\}$

(a) Determinar en cada caso una precondición P, la más débil que encuentre, de manera que se satisfaga la corrección de las anotaciones.

(b) El predicado P encontrado, ¿es *la precondición más débil*?

14. Considerar los siguientes programas que intercambian los valores de dos variables x e y de tipo Int:

$x, y := y, x$

$z := x;$
$x := y;$
$y := z$

$x := x - y;$
$y := x + y;$
$x := y - x$

(a) Especificar la pre y postcondición, y *verificá* los tres programas.

(b) Demostrar que el primer programa es equivalente al tercero, valiéndote de las siguientes propiedades de sustitución sintáctica en predicados:

- $(Q(x := E))(x := F) \equiv Q(x := E(x := F))$
  donde x es una (lista de) variable(s), y E y F son (listas de) expresiones bien definidas.

- $Q(x := E) \equiv Q(x, y := E, y)$
  donde x e y son variables distintas.

(c) ¿Es el segundo programa equivalente a los demás? En caso negativo, ¿como podemos relajar la definición de equivalencia, de modo que sea satisfecha por este programa respecto a cualquiera de los otros dos?

15. (a) Demostrar las siguientes equivalencias entre programas:

I. $x := x \simeq$ **skip**
II. S; **skip** $\simeq S$ y simétricamente **skip**; $S \simeq S$
III. S; **abort** $\simeq$ **abort** y simétricamente **abort**; $S \simeq$ **abort**
IV. $(S; T); U \simeq S; (T; U)$

(b) Pensando a ; como un operador binario, y haciendo analogía con las propiedades del cálculo proposicional, ¿qué nombres podríamos darle a las propiedades (2), (3) y (4)?